JAVA.UTIL.TREESET CLASS

http://www.tutorialspoint.com/java/util/java_util_treeset.htm

Introduction

The **java.util.TreeSet** class implements the **Set** interface.Following are the important points about TreeSet:

- The TreeSet class guarantees that the Map will be in ascending key order and backed by a TreeMap.
- The Map is sorted according to the natural sort method for the key Class, or by the Comparator provided at set creation time, that will depend on which constructor used.
- The ordering must be total in order for the Tree to function properly.

Class declaration

Following is the declaration for **java.util.TreeSet** class:

```
public class TreeSet<E>
   extends AbstractSet<E>
      implements NavigableSet<E>, Cloneable, Serializable
```

Parameters

Following is the parameter for **java.util.TreeSet** class:

- **E** -- This is the type of elements maintained by this set.

Class constructors

S.N.	Constructor & Description
1	**TreeSet()** This constructor constructs a new, empty tree set, sorted according to the natural ordering of its elements.
2	**TreeSet(Collection<? extends E> c)** This constructor constructs a new tree set containing the elements in the specified collection, sorted according to the natural ordering of its elements.
3	**TreeSet(Comparator<? super E> comparator)** This constructor constructs a new, empty tree set, sorted according to the specified comparator.
4	**TreeSet(SortedSet<E> s)** This constructor constructs a new tree set containing the same elements and using the same ordering as the specified sorted set.

Class methods

S.N.	Method & Description

1	boolean add(E e) This method adds the specified element to this set if it is not already present.
2	boolean addAll(Collection<? extends E> c) This method adds all of the elements in the specified collection to this set.
3	E ceiling(E e) This method returns the least element in this set greater than or equal to the given element, or null if there is no such element.
4	void clear() This method removes all of the elements from this set.
5	Object clone() This method returns a shallow copy of this TreeSet instance.
6	Comparator<? super E> comparator() This method returns the comparator used to order the elements in this set, or null if this set uses the natural ordering of its elements.
7	boolean contains(Object o) This method returns true if this set contains the specified element.
8	Iterator<E> descendingIterator() This method returns an iterator over the elements in this set in descending order.
9	NavigableSet<E> descendingSet() This method returns a reverse order view of the elements contained in this set.
10	E first() This method returns the first (lowest) element currently in this set.
11	E floor(E e) This method Returns the greatest element in this set less than or equal to the given element, or null if there is no such element.
12	SortedSet<E> headSet(E toElement) This method returns a view of the portion of this set whose elements are strictly less than toElement.
13	NavigableSet<E> headSet(E toElement, boolean inclusive) This method Returns a view of the portion of this set whose elements are less than (or equal to, if inclusive is true) toElement.
14	E higher(E e) This method returns the least element in this set strictly greater than the given element, or null if there is no such element.
15	boolean isEmpty() This method returns true if this set contains no elements.
16	Iterator<E> iterator() This method returns an iterator over the elements in this set in ascending order.
17	E last() This method returns the last (highest) element currently in this set.
18	E lower(E e)

	This method returns the greatest element in this set strictly less than the given element, or null if there is no such element.
19	E pollFirst() This method retrieves and removes the first (lowest) element, or returns null if this set is empty.
20	E pollLast() This method retrieves and removes the last (highest) element, or returns null if this set is empty.
21	boolean remove(Object o) This method removes the specified element from this set if it is present.
22	int size() This method returns the number of elements in this set (its cardinality).
23	NavigableSet<E> subSet(E fromElement, boolean fromInclusive, E toElement, boolean toInclusive) This method returns a view of the portion of this set whose elements range from fromElement to toElement.
24	SortedSet<E> subSet(E fromElement, E toElement) This method returns a view of the portion of this set whose elements range from fromElement, inclusive, to toElement, exclusive.
25	SortedSet<E> tailSet(E fromElement) This method returns a view of the portion of this set whose elements are greater than or equal to fromElement.
26	NavigableSet<E> tailSet(E fromElement, boolean inclusive) This method returns a view of the portion of this set whose elements are greater than (or equal to, if inclusive is true) fromElement.

Methods inherited

This class inherits methods from the following classes:

- java.util.AbstractSet
- java.util.AbstractCollection
- java.util.Object
- java.util.Set